IN THE SPECIFICATION

Please amend the paragraph that starts at page 4, line 5 as follows:

The part of the state data structure that is related to one set of data packets is called an entry. When a set of data packets has been handled, e.g. packet data connection ~~is~~ has been closed, the corresponding entry is cleared in the state data structure.

Please amend the paragraph that starts at page 4, line 27 as follows:

An implementation better suitable for handling the state information within a network element cluster is that each node maintains a state data structure containing state information used for handling data packets handled by any node of said cluster, i.e. state information relating to all sets of data packets or connections handled in said cluster. Each node adds new entries/clears old entries in its own state data structure as it handles data packets and communicates information about said new and old entries to other nodes of the cluster corresponding entries to be added/cleared in their state data structure. This communicating of information may be done e.g. on timely basis.

Please amend the paragraph that starts at page 5, line 1 as follows:

Typically there are maintained two identical state data structures in nodes described above. One is typically maintained in kernel space. This may be called an active state data structure, since data packets are handled using this state data structure and new entries are added/old entries cleared in that state data structure. The other one is in practice a duplicate of the state data structure in kernel space and is maintained in user space. The entries added/cleared in the kernel space data structure are updated to the user space data structure. The user space state data structures are typically maintained for synchronising purposes. That is, entries of the user space state data structures are communicated between nodes of the cluster. Typically information about added/cleared ~~added cleared~~ entries are communicated and user space state data structures of other nodes are updated accordingly. The changes in the user space state data structures are then pushed to the kernel space state data structures of respective nodes. This way both user space and kernel space state data structures contain information about all sets of data packets handled in the cluster and transferring connections between nodes is reliable, since information about the sets of data packets is readily maintained in kernel space in all nodes.

Please amend the paragraph that starts at page 5, line 17 as follows:

An example of a network element cluster CA in accordance with the discussion above is illustrated in Figure 1B. Three nodes Node1, Node2 and Node3 or respectively A1, A2 and A3 of the network element cluster CA and the state data structures in the nodes are illustrated. The network element cluster CA may be for example the network element cluster CA of Figure 1. In each node Node1, Node2 and Node3 there are maintained respectively active data structures 11b, 12b and 13b, which are used for handling data packets, and additional state data structures 11a, 12a and 13a for synchronizing the state data structures of the nodes with each other. The nodes Node1, Node2 and Node3 are provided with ~~possibility~~ the ability to communicate information between the state data structures of other nodes and between their own state data structures. All state data structures change dynamically in time responsive to adding new entries or clearing old entries in the state data structure of any one of the nodes. Effectively all state data structures 11-13 have identical contents including entries related to all sets of data packet handled in the cluster at a given moment of time.

Please amend the paragraph that starts at page 7, line 3 as follows:

In a node there is thus maintained two different state data structures: a first, node-specific data structure comprising entries representing state information needed for handling sets of data packets handled in said node, and in addition to said node-specific data structure a second, common data structure comprising at least entries representing state information needed for handling sets of data packets handled in one other node of said network element cluster. The contents of said common data structure effectively differs from the contents of said node-specific data structure, and the contents of the node-specific data structure is changed in the course of time to correspond to the sets of data packets handled in said node at any given moment of time.

Please amend the paragraph that starts at page 9, line 6 as follows:

The packet data connections discussed here are typically packet data connections on IP protocol. In this specification and in the appended claims, the term "packet data connection" ~~packet data connection~~ refers here to a bi-directional flow of data packets. Examples of such packet data connections are TCP connections, bi-directional UDP packet flows, UDP queries, ICMP queries and replies, and connections according to various protocols operating on top of TCP or UDP.

Please amend the paragraph that starts at page 9, line 6 as follows:

In this specification and in the appended claims, the term entry refers to a piece of information relating to one set of data packets. An entry typically comprises information at least about data packet headers. The term "set of data packets" ~~set of data packets~~ on the other hand refers to data packets, which are related to each other, such as data packets of a packet data connection, data packets of a communication session comprising a plurality of packet data connections, data packets of a plurality of packet data connections of a secure tunnel, or any other suitable set of data packets. The terms "state data structure", "node-specific data structure" or "common data structure" ~~term state data structure, node specific data structure or common data structure~~ refer to a data structure, whose entries represent sets of data packets handled in a network element. Such data structures may be, for example, a table or a linked list or any other more versatile data structure.

Please amend the paragraph that starts at page 12, line 11 as follows:

In step 212 in a plurality of entries of said node-specific and common data structures is maintained distribution information relating to the distribution identifier, which corresponds to the set of data packets related to the respective entry. This step makes it ~~enables that it is~~ possible to identify to which node a particular entry in the node-specific and common data structures may belong ~~belongs~~ at a given moment in time. This is a key element for making it possible to implement the invention. Since there is knowledge of the grounds of distributing sets of data packets to different nodes, it is possible to keep track ~~on~~ which entries belong to which node even if distribution of the sets of data packets varies dynamically.

Please amend the paragraph that starts at page 13, line 13 as follows:

Figure 4 illustrates as an example a flowchart of a method adding entries to the state data structures. In general, a new entry representing state information related to a set of data packets is added to a state data structure as a first data packet of the set of data packets is handled. The state data structure entry is then used for handling the other data packets of the set of data packets. After handling all data packets of the set, ~~set are~~ the corresponding entry is cleared from the state data structure. It is well known how to determine which data packets cause adding a new entry and which are handled according to the state data structure. For

example, a data packet may be compared against entries in a state data structure and if a corresponding entry is not found a new entry needs to be made. If a corresponding entry is found the data packet is handled accordingly. Additionally, it is possible to perform a further check whether the new entry would be allowable e.g. on the basis of rules, before adding the new entry in the node-specific data structure. Furthermore, some data packets may require special handling and/or special entries in the state data structure. For example a specific separate code portion may be required for handling some set of data packets and such code portion may create to the state data structure new entries for some other set of data packets on the basis of data packets it is handling. Creating an entry for FTP data connection on the basis of related FTP control connection is an example of such handling of data packets. Such handling of data packets is however beyond the scope of this invention and is not addressed here any further. Considering the invention it is irrelevant on what basis an entry is added or cleared in a state data structure, since the invention does not concern determining the entries, but rather handling existing entries.

Please amend the paragraph that starts at page 16, line 35 as follows:

The nodes Node1, Node2 and Node3 are provided with the ability to ~~possibility~~ communicate information between the common data structures of other nodes and between their own common and node-specific data structures. The common data structures change dynamically in time responsive to adding new entries or clearing old entries in the node-specific data structures due to handling sets of data packets as was explained in more detail in connection with Figure 4. Whereas node-specific data structures change dynamically in time due to handling sets of data packets, and also due to reallocation of distribution identifiers as is explained in more detail in connection with Figure 6. The state data structures of the nodes are presented as examples only and it should be clear that it is possible to implement the state data structures also in some other way within the scope of the invention.

Please amend the paragraph that starts at page 16, line 35 as follows:

In order to clarify the operation during reallocation of distribution identifiers let's consider a situation where distribution identifiers 1, 2 and 3 are initially allocated to a node 1 and distribution identifiers 4 and 5 to a node 2. Due to load balancing the distribution identifier 3 needs to be reallocated to the node 2, and consequently the sets of data packets relating to the distribution identifier 3 need to be

transferred to the node 2. The node 2 receives information about the reallocation and thus the entries corresponding to the distribution identifier 3 are searched for in the common data structure of the node 2 and the corresponding entries of the common structure are added to the node-specific data structure of the node 2. Then distribution identifier 3 is actually reallocated to the node 2, i.e. the data packets relating to the distribution identifier 3 ~~are~~ begin to be distributed to the node 2 instead of the node 1, and node 2 continues to handle the corresponding sets of data packets. After this the entries corresponding to the distribution identifier 3 are cleared from the node-specific data structure of the node 1. It is possible that there is a short break in handling the data packets arriving at the nodes during which none of the nodes owns the distribution identifier 3, but it is not likely that any connections fail because of this very short break and therefore this break or delay can be considered negligible.